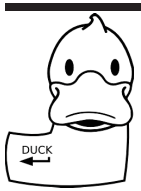

DUCK : a deDUCtive Keyboard

Philippe Roussille
IRIT – Elipse
Université de Toulouse
31062, Toulouse cedex 9
France
philippe.roussille@irit.fr

Emmanuel Dubois
IRIT – Elipse
Université de Toulouse
31062, Toulouse cedex 9
France
emmanuel.dubois@irit.fr

Matthieu Raynal
IRIT – Elipse
Université de Toulouse
31062, Toulouse cedex 9
France
mathieu.raynal@irit.fr

Christophe Jouffrais
IRIT – Elipse
CNRS & Université de Toulouse
31062, Toulouse cedex 9
France
christophe.jouffrais@irit.fr

Slim Kammoun
IRIT – Elipse
Université de Toulouse
31062, Toulouse cedex 9
France
slim.kammoun@irit.fr

Abstract
This paper presents the deDUCtive Keyboard (DUCK), aiming to improve text entry for visually impaired users on AZERTY/QWERTY based layout on software keyboards. Relying on a predictive system, DUCK allows rapid text entry without any precision on keyboard hits. A preliminary study with a visually impaired user indicated that usability is improved when compared to a regular virtual keyboard with a vocal feedback.

Author Keywords
accessibility, input method, keyboard, mobile device, software, text entry, visual impairment

ACM Classification Keywords
H.4.2 [Computers and Society]: Social Issues - Assistive Technologies for Persons with Disabilities; H.5.2 [User Interfaces]: Information Interfaces and Presentation

General Terms
Accessibility

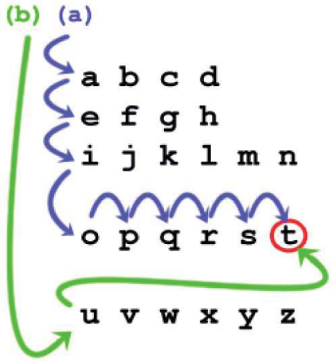

Figure 1: Alphabet with NavTap

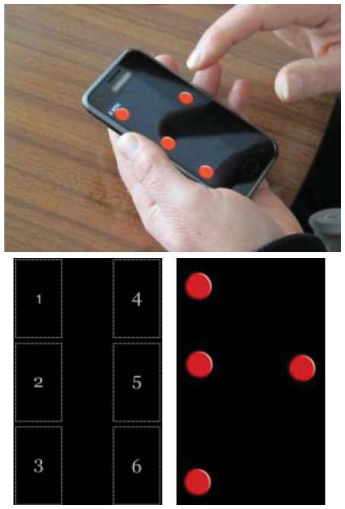

Figure 2: BrailleType dot matrix

Introduction

During the last years, mobile devices have been equipped with touchscreens, additional technologies (WiFi, GPS, etc.) and new interactions techniques. These devices which were originally used only to make calls, have evolved into full connected tools. They are used to exchange short text messages[1] , communicate using instant messaging protocols, or deeply interact on a wide scale with complex social networks (Facebook, Twitter, etc.) and Internet.

As a result, text entry on such devices has become a critical task. As the main trend goes with touchscreens, software based techniques are more and more popular. It is obvious that touchscreens are more flexible than physical keypads to design dynamic interfaces; but at the expanse of a huge drawback : accessibility. Indeed, visually impaired people lack tactile clues to locate the keys when typing.

In the following, we first present different techniques which have been proposed to restore accessibility of virtual mobile keyboard. We then present a new technique called DUCK, as well as the results of a preliminary evaluation with a blind person. Finally, we open the discussion towards possible future works.

Related works

The most common and used methods to assist text entry on mobile devices consist on using Text To Speech (TTS) synthesis with a standard keyboard layout (e.g. QWERTY or AZERTY, depending on the culture). When the user moves his finger onto the keyboard or hits a key, the keys he hits or he moves across are spoken (see e.g. Apple VoiceOver). The user can also use additional gestures to perform text entry. However, the keys of a virtual keyboard on a mobile device are very small. Thus, the risk of mistyping a letter is quite high. The second hindrance is the need for users to go through most of the keys before finding the one they seek (known as "*painful exploration*"), which usually takes some time. The offered sets of gestures might also be too complex to perform in mobility.

Another category of text entry method especially designed for visually impaired users is based on the Braille alphabet (see e.g. BrailleType[6] – cf. Fig. 2) & TypeInBraille [5] – cf. Fig. 3). They mimic a 6-dots Braille cell onto the phone screen. The characters are entered by successively activating the dots of the cell. Unlike the QWERTY standard keyboard, BrailleType proves to have a much easier learning phase, due to the similarity with the Braille alphabet. However, the user must fill-in the dot matrix for each character, which leads to a quite high Keystroke Per Character (KSPC), hence drastically reducing text-entry speed.

NavTouch & NavTap[1] (*cf.* Fig. 1) are still different text-entry methods for the visually impaired. The user can dynamically select a letter with a specific gesture performed anywhere on the screen, drawn on the device, (location-independent on the keyboard). These techniques are very efficient for quickly targeting one letter. However, they generate high cognitive load to simultaneously remember the correct gesture, as well as the position of the typed letter in the word.

Some techniques, such as BlindType, Swype and SHARK[4] heavily rely on character prediction and user typing anticipation, enhancing their dictionaries through constant use. These systems allow the user to dissociate rapid typing and precision typing, using the keyboard

[1] ABI_Research, More than seven trillion SMS messages will be sent in 2011.

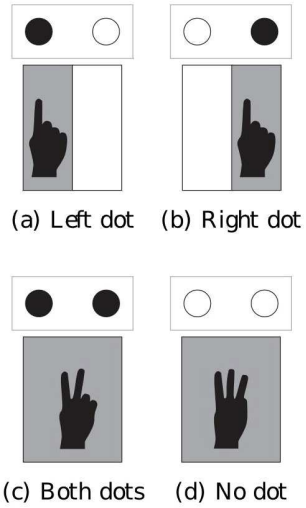

Figure 3: TypeInBraille finger gestures

letters as mementos and landmarks for typed or drawn patterns. In spite of allowing fast typing, these keyboards are not made accessible for the blind (low or no nonvisual feedback).

The No-Look-Notes[2] text-entry method uses multiple fingers, relying on the multi-touch capacities of the device. The screen is divided in different zones, with a set of characters associated for each. The user can search for a character among a group of letters with one finger, and confirms the selection via a split-tapping. This method provides a fast and easy access to letters, but drastically increases the number of taps, which slows word typing down, and may cause muscular fatigue.

DUCK: the deDUCtive Keyboard

DUCK Principle

We designed DUCK (deDUCtive Keyboard) based on either the AZERTY or QWERTY layouts. Actually these layouts are the most used ones, meaning that most people are acquainted with them. We decided, however, to simplify the user interaction to type letters. The main objective was to get rid of the accurate search of each character on the layout. Besides, we aimed to compensate for the lack of precision related to small and mobile keyboards.

Design

As DUCK targets visually impaired users, we designed a full-screen keyboard, preventing calling sub-menus or misinterpreted keystrokes. The user initially explores the keyboard to find the first key of the intended word. Each finger motion provides a vocal feedback, allowing the user to locate a letter on the layout. Once he releases the key, the corresponding character is selected. For typing the remaining letters of the word, the user doesn't have to explore the keyboard again. He just needs to press the location where he believes the keys are according to the memorized representation of the AZERTY or QWERTY layout. The user signals the end of the word by pressing two fingers on the screen. Different chimes describe the keyboard state. Typed words are finally spoken entirely to avoid confusion.

Implementation

DUCK relies on a different approach than the usual prediction technique. Instead of offering possible words starting with what the user is typing, DUCK waits for a full word (two fingers press). It then detects and corrects the possible typing errors, similarly to SHARK[4]. DUCK analyses the keystrokes that were made. A first filter is applied to select a list of words beginning with the same first letter and the same length. DUCK then computes the distance between the center of the keys corresponding to each letter of these words, and the successive locations hit by the user. Finally, given the distances, the DUCK returns the best five matches, which are the words minimizing the sum of these distances. The user can cycle through these five proposition with a one finger press, and can validate his choice with a two fingers press.

Modes

The commands in the DUCK are designed to be simple and easy to use without visual feedback, as they do not require complex strokes[3]. In addition, the keyboard has four modes that can be toggled by simple gestures:

a. the letter deductive mode (which is the default mode);
b. a numerical input mode to type numbers and operators (toggled by a two fingers up swipe);
c. a symbol input mode for punctuation and non-alphanumerical characters (toggled by a two

fingers down swipe);

d. a spelling mode allowing the user to add a word to the dictionary (toggled by pressing with three fingers on the screen)

At any moment, the user can insert a space by swiping the screen right with two fingers, or remove the last keystroke/word/character by swiping the screen left. The user can switch back to the previous mode by doing the opposite gesture. Our objective was to design simple but distinctive gestures; thus we preferred the use of multi touch to a set of single finger strokes[3].

Case study

Procedure

We designed a case study with one blind user without additional sensory or cognitive impairment, accustomed to a daily use of a computer and a mobile phone. We compared the speed and accuracy of the DUCK versus a mobile keyboard augmented with VoiceOver. Two sessions were conducted. Both sessions were done on a Samsung Galaxy S, running Android 2.3.6 as the operating system. In each session, the participant had to execute two separate typing tasks using either DUCK or VoiceOver. The first task consisted in typing a list of 30 different French words. The second task comprised typing 15 different short sentences. The participant was allowed to rest between each task. As the user does not have to scan through the entire keyboard between each character input, we made the main hypothesis that DUCK is faster than a VoiceOver like keyboard.

Results

Although there was quite an observable difference between the number of characters per second (CPS) during the two tasks of a same session – the user achieved 0.42 CPS with the VoiceOver like keyboard versus 0.53 CPS with DUCK (25% increase) – we had too few participants to make this observation significant. The exploration time was about three times shorter for DUCK compared to VoiceOver (1004,5 ms vs. 2920,2 ms). The error rate per character for the VoiceOver keyboard was 16.1%, while the error rate per word for DUCK was 23.5%. With DUCK, the words were validated without any correction in 81.5% of the cases. In average, predicted words were at the 1.27 position in the list. They were in the 1st proposition in 74% of the cases.

Discussion & Future work

The preliminary evaluation showed that DUCK highly reduced the time between keystrokes. It also provided very accurate output in the vast majority of the situations. The only negative result concerned the error rate. This was an expected result, as a mistyped word in DUCK forces the user to retype the whole word. We are currently improving the presentation of the predicted words, in order to decrease the error rate. We also noted two points where we can improve DUCK. On the one hand, we want to decrease the latency caused by the text-to-speech feedback. Indeed, the user is slowed down as he has to wait for his input to be spoken so he can check what he entered. On the other hand, we are designing a method to distinguish homophone words. On the other hand, we could improve our prediction results greatly by using a similar prediction system such as SwiftKey, where we could modify the suggestions order on behalf of the user's habits. Finally the results may be different according to the touch-screen size (e.g. a smartphone vs. a tablet) and to the length and type of the document (i.e. a short text vs. a more structured document). We are planning an experiment with more participants and conditions in order to assess the significance and validity of the results.

References

[1] Bonner, M. N., Brudvik, J. T., Abowd, G. D., and Edwards, W. K. No-look notes: accessible eyes-free multi-touch text entry. In *Proceedings of the 8th international conference on Pervasive Computing*, Pervasive'10, Springer-Verlag (Berlin, Heidelberg, 2010), 409–426.

[2] Guerreiro, T., Lagoá, P., Nicolau, H., Gonçalves, D., and Jorge, J. A. From tapping to touching: Making touch screens accessible to blind users. *IEEE MultiMedia 15*, 4 (Oct. 2008), 48–50.

[3] Kane, S. K., Bigham, J. P., and Wobbrock, J. O. Slide rule: making mobile touch screens accessible to blind people using multi-touch interaction techniques. In *Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility*, Assets '08, ACM (New York, NY, USA, 2008), 73–80.

[4] Kristensson, P.-O., and Zhai, S. Shark2: a large vocabulary shorthand writing system for pen-based computers. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, UIST '04, ACM (New York, NY, USA, 2004), 43–52.

[5] Mascetti, S., Bernareggi, C., and Belotti, M. Typeinbraille: quick eyes-free typing on smartphones. In *Proceedings of the 13th international conference on Computers Helping People with Special Needs - Volume Part II*, ICCHP'12, Springer-Verlag (Berlin, Heidelberg, 2012), 615–622.

[6] Oliveira, J. a., Guerreiro, T., Nicolau, H., Jorge, J., and Gonçalves, D. Brailletype: unleashing braille over touch screen mobile phones. In *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part I*, INTERACT'11, Springer-Verlag (Berlin, Heidelberg, 2011), 100–107.